An Agent Solution to Flexible Planning and Scheduling of Passenger Trips

Claudio Cubillos[1] and Franco Guidi-Polanco[2]
1 Escuela de Ingeniería Informática, Pontificia Universidad Católica de Valparaíso, Av. Brazil 2241, Valparaíso, Chile
claudio.cubillos@ucv.cl
2 Escuela de Ingeniería Industrial, Pontificia Universidad Católica de Valparaíso, Av. Brasil 2241, Valparaíso, Chile
fguidi@ucv.cl

Abstract. This work presents the MADARP agent architecture, devoted to the planning and scheduling of trip requests under a dynamic scenario within the context of passenger transportation systems. The architecture provides a set of base agents that perform the basic interface, planning and support services for managing different types of transportation requests by using a heterogeneous fleet of transport vehicles. The architecture was used to implement three planning models by extending base agents' behaviours. The results obtained for a set of 20 scenarios is analyzed.

1 Introduction

The field of passenger transport systems has received an increasing attention as citizens require more flexible transportation alternatives in their cities. As response, new alternatives to satisfy the transport demands of citizens are being conceived [1].

Changes in transport requirements in European citizens have brought the opportunity to create new services aimed to fulfill special transportation demand in addition to regular population mobility services. Their objective is to satisfy personal transportation requests at relatively low costs, thanks to an integrated planning with the use of the different available resources on transport networks.

From a technological perspective, the recent advances in network systems together with the low cost of the processing power have move us to the era of distributed systems and ubiquity. In this trend, the integration, transparency and interoperation among heterogeneous systems are a must. Hence, the multiagent paradigm [16] appears as a promising technology, capable of tackling these newer requirements in an efficient and sustainable way.

This work presents the MADARP architecture, devoted to the implementation of flexible passenger transportation systems. It provides agents which implement the basic planning and scheduling functionality for processing transport requests coming from different kinds of users and by considering a heterogeneous fleet of transport vehicles.

2 Transportation Requirements

From a mathematical point of view the transportation problem involved corresponds to the dynamic version of the Dial-a-ride Problem (D-DARP) known to be NP-hard. For this reason all of the commercial solutions and most of the research are focused in heuristic solutions.

Clients commonly specify transport requests with a pick-up and delivery place. They also indicate time windows, that is, time intervals within which the client has to be picked-up at the origin node and delivered at the destination node. Moreover, the requests can include further descriptions of the desired service like type and number of places, shared or exclusive use of the vehicle, wheelchair place use and any other complementary services.

Besides, the passenger transportation system we are tackling considers heterogeneous fleets of vehicles, composed by busses, minivans, vehicles for disabled people, taxis, among others. These vehicles may have diverse characteristics such as: limited passenger's capacity and availability time-periods along the day, an specific area to cover, types of seats, low floor, wide access, no stairs or complementary services like Bar, WC, air conditioning and bicycle transport among others. These properties usually affect the client's comfort and consequently their perception of the received transport service.

In addition, a dynamic scenario is considered, in which the vehicle progress is monitored; clients can modify or cancel their trip requests, vehicle delays can occur, clients may not show up at the pickup place and vehicles can breakdown, all of them involving the re-scheduling of the trips and their management.

2.1 Related work

A software system for D-DARP was proposed by Horn [7]. The optimization capabilities of the system are based on least-cost insertions of new requests and periodic re-optimization of the planned routes. Finally, Coslovich et al. [4] have addressed a dynamic dial-a-ride where people might unexpectedly ask a driver for a trip at a given stop by using a two-phase method and a neighborhood of solutions generated off-line.

Newer research tackling the dynamic problem tends to use a distributed market-based philosophy based in the Contract-Net Protocol (CNP) (see [3], [5], [6] and [11]). The MARS System [6] and the TeleTruck [3] approach use the Extended Contract-Net Protocol (ECNP) with Simulated Trading improvement for dealing with dynamics and uncertainty in a transportation scheduling problem.

Soft computing has also been applied to the transport domain with the use of genetic algorithms (GA) for the optimization of the assignment (see [8] and [15]) and systems based on an ant-colony as reported in [12]. Teodorovic and Radivojevic [14] have later studied a generic version of the dynamic DARP using fuzzy logic for the travel times, as well as Kikuchi and Donnelly [9].

Finally, agent-based systems are presented in [5], [10] and [13]. All of them make use of the CNP for the assignment of client's rides. In addition, [10] uses a stochastic post-optimization phase to improve the result initially obtained. It works in a similar way to the simulated trading. In [13] is presented the Provisional Agreement Protocol (PAP), based on the ECNP and de-commitment. Its improvement is to allow biddings for partial routes and overcomes the Eager Bidder Problem of the CNP, that is, the contractor commits to the bid even though the bid has not been granted and hence cannot make the same bid to another.

3 The Agent Architecture

The agent architecture is built-up over the Jade agent platform [2], which provides a distributed environment organized in containers where agents can work, communicate and migrate within them. Figure 1 shows the MADARP agent architecture [5] which shows four layers that group the agents and structures according to the functionality provided. The Interface layer connects the system with the real world; the Planning layer performs the trips processing; and the Service layer provides different complementary functionalities. At the bottom the Service Ontology provides a means to integrate and make interacting the different agents and actors from the upper layers in a transparent and coherent way.

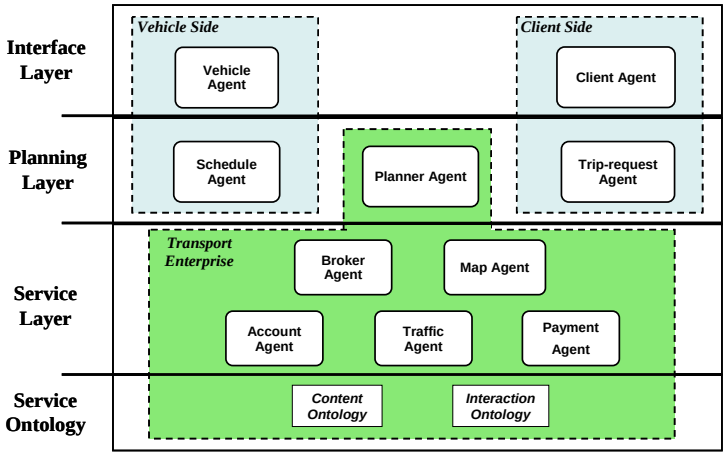

Fig. 1. The multiagent transportation architecture.

Figure 1 shows also the three main actors involved in the transportation chain: vehicles, clients and the transportation enterprise; each of them modeled in terms of

agents. Consequently, each vehicle actor is represented by a Vehicle agent and a Schedule agent. In a similar way, each client is characterized by a Client agent and a Trip-request agent. In both cases, the pair of agents is tightly coupled as they are modeling different aspects of the same real entity. The third actor is the transport enterprise, which is built up by a series of agents and structures that provide support to diverse services related with the planning and control of the passenger transportation service provided.

The routing and scheduling functionality provided by the architecture is based on the contract-net protocol (CNP) plus a possible negotiation phase. The interaction among the planning agents is as follows (see Figure 1): First, each transportation request coming from a Client is received by the corresponding Trip-request agent of the couple, which asks the Planner to process it. Next, the Planner processes the request first by obtaining from the Broker agent the vehicles that match the required profile, and then by making a call for trip-proposals to all the corresponding Schedule agents (call for bids in contract-net) that represent the different vehicles of the considered fleet. They send back their proposals and the Planner selects the most suitable alternatives among the received trip proposals by applying filters and starts a negotiation process with the client (through its Trip-request agent). After arriving to agreement the Planner tells the Schedule agent that won the proposal to add the trip to its actual schedule and tells the others their proposal rejection.

Upon differences in the planning (due to breakdowns, traffic jam, etc) the Schedule agent re-plans. In the case of having an infeasible trip request (mainly due to the time-window restrictions), it informs the Planner agent about the situation. The Planner makes a call for trip-proposals to try reallocating the request in other available vehicle. In any case, the result is informed to the corresponding Trip-request agent, which depending on its degree of autonomy will process the alternatives and take a decision or will inform the client about the change. This change may imply a different vehicle processing the trip only or also a delay or an anticipation of the pickup and delivery times defined previously. This default planning implementation can be modified or extended by overwriting the set of behaviours of the different base agents, which are detailed in the following.

3.1 Client Side

Individual clients and their requirements are captured by Client and Trip-request agents. Together they provide full communication and interoperability of the real end user with the transportation system. The Client agent is in charge of providing a personalized user interface while the Trip-request manages the process of requesting the service, its characteristics and the decision making required.

The interface provided by Client agents should be adaptable to the different devices (cell phones, PDAs or PCs). In addition, the Client agent is responsible for capturing all the client's requirements not only concerning the desired type of transport service but also his preferences upon contingency situations (e.g. delays, traffic jams, deviations, etc). The Trip-request takes these requirements and preferences to act on behalf of the real client during the whole process. Depending

on the degree of autonomy provided by the client, the Trip-request can act as a personal trip assistant or simply as a mere proxy of the client decisions.

This agent contains three base behaviours. The Schedule_me_Behaviour is a one-shot behaviour which in its default implementation is in charge of receiving the desired transport service from the real client through the client agent that acts as interface. The message contains the request profile that is then forwarded to the planner inside a trip request message.

The Negotiate_Behaviour processes the subsequent messages coming from the planner in order to arrive to an agreement. This depends on the underlying negotiation protocol implemented by both, the Trip-request and the planner agents. The default implementation for this behaviour receives a list with filtered proposals which are evaluated by using the client's utility function.

The Send_status_Behaviour is a cyclic behaviour performing a utility service. It turns back to the sender the status of the request being treated.

3.2 Vehicle Side

Each real vehicle is represented by an agent couple, the Vehicle and the Schedule agents. They provide interoperability between the vehicle they represent and the transportation system to which they belong to. The Vehicle agent plays an interface role, providing the vehicle and its driver with a communication channel with the rest of the transportation system. Through it, the driver is able to communicate along the journey about any contingency that could arise. The Schedule agent is in charge of managing the vehicle's route and processing any new request for client transportation.

The Vehicle agent contains two behaviours. The Register_Behaviour performs the registration of the vehicle with the broker agent. Therefore, it sends a subscribe message to the broker containing a Service Profile.

The Inform_event_Behaviour is the core behavior as it enables the agent to inform about the status of the vehicle and its route advancement. It sends an inform message to its corresponding Schedule agent containing an event description, such as the vehicle arrival to a pickup/delivery place, the presentation or no presentation of the client at the predefined stop, a vehicle malfunction, an emergency, the vehicle deviation due to an accident, traffic jam, etc.

Base Schedule agents implement two cyclic behaviours. The Evaluate_trip_Behaviour evaluates the insertion of a trip (client) in its current schedule. The default implementation listens to the calls-for-proposals coming from the Planner. In case of not being committed by that time with other call (not already finished) it will prepare a proposal, otherwise it will answer back with a refusal message. When preparing the proposal (profile), the behaviour decodes the client's Request Profile description attached in the call's content and evaluates the trip inclusion in the vehicle route.

The Wait_proposal_answer_Behaviour is much coupled to the previous one, as it process the planner's answer with respect to the formulated proposal. For this, the default behaviour checks the planner's message to see if the proposal has been accepted or rejected. In case of an accepted proposal, the behaviour uses the generic interface to insert the trip in the vehicle's route.

3.3 Transportation Enterprise Side

The transportation service role is mainly carried out by the Planner agent acting as a front face to Trip-requests and Schedule agents. The Planner has seven behaviours.

As it name says, the Process_request_Behaviour processes the incoming Schedule-me messages of Trip-request agents. The agent creates a registry of the new request to add to its list. It also decodes the Request Profile contained in the incoming message, and sends a query message to the broker asking for the vehicles that match the requirements contained in the Request Profile.

The CallForProposals_Behaviour receives the broker's answer containing the list of agents that match the desired service described in the Request Profile. The behaviour decodes the list and sends a call-for-proposal message to all the corresponding ScheduleAgents contained in the list. In case of a failure message from the broker, the behaviour will forward it to the corresponding Trip-request agent.

The Process_proposal_Behaviour receives the answer messages from the ScheduleAgents. These messages carry in their content the trip proposals (Proposal Profiles). The behaviour checks if all ScheduleAgents have answered back to the call. If that is the case, then a StartNegotiation_Behaviour is instantiated and activated in the agent.

The StartNegotiation_Behaviour is a one shoot behaviour that can start in two ways; by a Process_proposal_Behaviour that received the last pending answer to the call or by a Request_timeout_Behaviour triggered by the call deadline. The behaviour gets all the proposals received for a given trip request and applies them the filters (if any) contained in its policies list. This will result in discarding some of the proposals. After the filter process, the behaviour starts the negotiation procedure with the Trip-request agent by sending a message to it.

The Process_client_choose_Behaviour also depends on the implemented negotiation protocol. The behaviour receives the Trip-request answers and sends counter proposals until a deal is obtained or the protocol finishes. In any case, the result is forwarded to the involved ScheduleAgents by sending an accept proposal message to the winner and a reject proposal message to the rest.

Finally, the Process_schedule_confirm_Behaviour ensures that the winning ScheduleAgent has committed to the transport request. The default behaviour receives the winner’s answer and forwards it to the corresponding Trip-request agent.

Besides the Planner Agent, there is a whole set of service agents collaborating to give support to the different required functions, such as the matching of request to vehicles, the geographical data access, the accountability of the transactions and the service payment among others. From them, the most critical ones from the planning and control point of view are the Broker and the Map agents.

The Broker is the one in charge of carrying out the matching of transport requests with available vehicles. For that is able to manage service descriptions coming from both sides, understand their semantics and perform the search.

The Map agent represents the geographic area being considered where it can be a zone, a city or a part of it. The Map provides the enterprise with a series of

information regarding the actual zone being covered such as localization of addresses and stops, street names and distances between localizations, among others.

4 Concrete Planning Systems

Three models of transport planning system were implemented in order to test the agent-based architecture. These were: a centralized, a market-based (decentralized) and a mediated one. For each model, the architecture's base agents were extended and modified. These are explained in the following.

The *Centralized Model* (C) considers the optimization of the global utility for the system. It pursues the minimization of a disutility function of the fleet operator (number of vehicles required, fixed and variable travelling costs) and the served users (effective waiting time, effective ride time). For this to happen, the behaviour of Schedule agents is overwritten in order to consider the vehicle utility function plus the client's one.

The implemented *Decentralized Model* (D) is an approach based in the contract-net (CNP) under self-interested agents. Therefore, Vehicle agents pursued the optimization of the travelling costs (utility function with total slack time and total travel time) and Client agents were oriented towards the maximization of the perceived service quality (utility function with excess travel time and waiting time).

The *Mediated Model* (M) takes advantage of the mediation role of the Planner by filtering the received proposals. The mediated model involves a two-phase planning: First, a Call-For-Proposals (CFP) started by the Planner and answered by the Schedule agents and then a negotiation process that pursues an agreement between the Planner and the client. This two-phased model offers a major difference; the Planner with its filtering and negotiation policies performs a mediation role. It implements the partial centralization of this mixed approach, getting solutions closer to the global optimum when compared with complete decentralized models.

In practical terms the steps are implemented as in the previous approach (the decentralized one). It only changes the Planner role. In this implementation the Planner does apply a filtering policy to the list of received proposals.

4.1 Results

Here are presented some of the results obtained with the 3 concrete models. All the tests considered the same geographical net and 20 demand scenarios with 50 trip requests each, distributed uniformly in a two-hour horizon. For each demand scenario 25 runs were done.

The tests on the three models (see Figure 2) have shown that on average the Mediated model is able to provide results in the gap between the Centralized and Decentralized models.

In general terms, the decentralized model will provide better results for the clients, in terms of both, Waiting Time and Excess Ride Time as these are variables that measure the solution quality from the clients' perspective. In fact, when looking at Figure 2 the Decentralized values are the lowest almost for all the scenarios, while

the Centralized values correspond to the highest ones. This is explained because this model provides solutions that are better for vehicle operators and worse for clients.

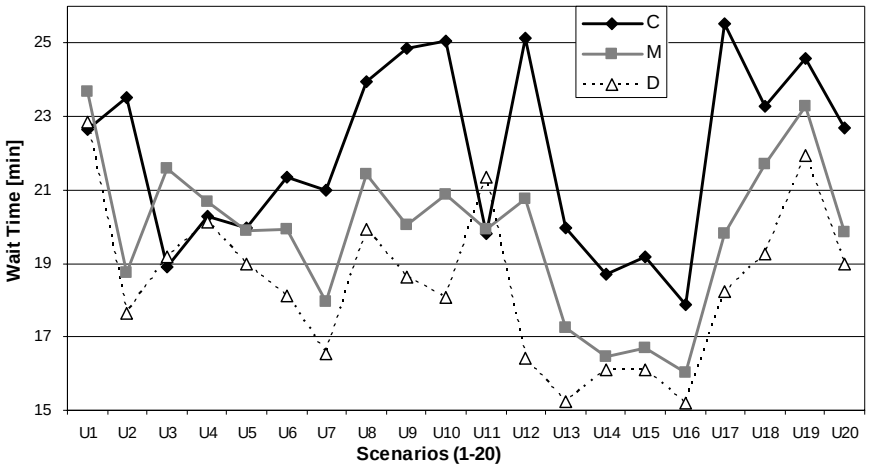

Fig. 2. Graph showing the wait time obtained for the 20 scenarios under the three planning models.

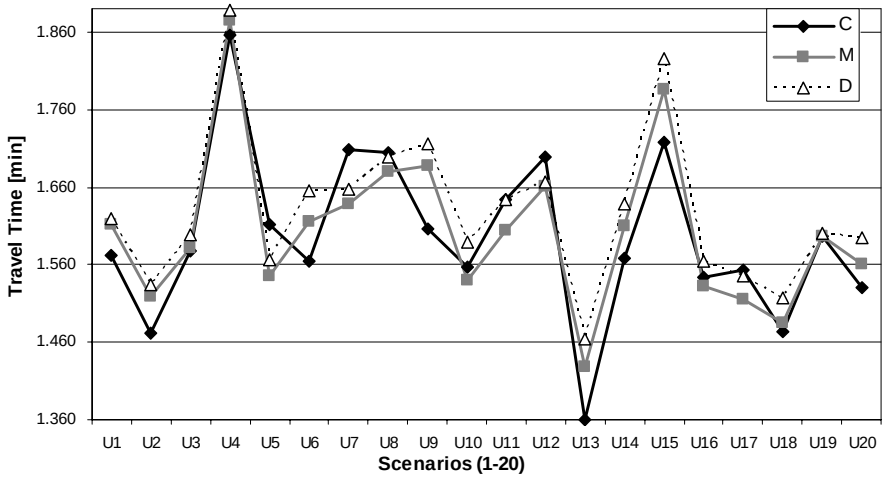

Fig. 3. Graph showing the travel time obtained for the 20 scenarios under the three planning models.

In the same Figure 2 we can appreciate that the Mediated model is just in between, providing almost in all scenarios a middle value for the wait time measure. A similar thing happens with the other client measure considered in the tests, the excess ride time.

As already mentioned, the Centralized approach tends to provide results that in terms of preference, tend to benefit more the vehicle operators rather than clients, which is just the opposite way for the decentralized model. Therefore, when

comparing the results of values regarding performance measures of the vehicles' operators, the situation is inverted. For example, in the case of the travel time, the values corresponding to the Centralized model are the lowest for almost all cases as Figure 3 shows. In fact, within the 20 considered scenarios in only five occasions the centralized was the highest among the three alternatives.

In a similar way, the Decentralized approach behaves providing the highest values for the travel time. Again in most scenarios (16 out of 20) the travel time from the decentralized model was the highest among the three alternatives. The interesting thing is that the Mediated model provides results just in between or even below the others in all 20 scenarios. Analogous results are obtained for the other operators' performance measure considered in the test, the slack time.

Finally by looking at Figure 4 that shows the average total cost of the three models on the 20 scenarios, it is possible to see that the Mediated model provides in-between or lower costs in all the 20 scenarios.

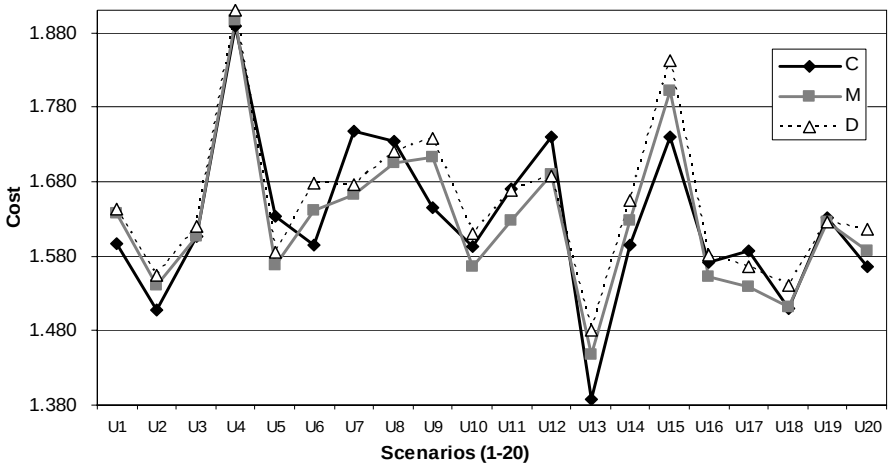

Fig. 4. Graph showing the cost obtained for the 20 scenarios under the three planning models.

To sum up, the analysis shows that the Mediated model gives better results for the clients, in terms of both, Waiting Time and Excess Ride Time when compared with the Centralized model. On the other hand, when compared with de Decentralized model, it gives better results from the vehicle's (or operator's) viewpoint, this in terms of Travel Time and Slack Time.

5 Conclusions

The MADARP architecture for implementing agent-based passenger transportation systems was described. The layered model enforces the system reutilization and maintainability, as more low level and general services (provided by JADE) are decoupled from the domain specific ones (base agents plus ontology). By extending the base agents is possible to obtain an ad-hoc concrete planning system. MADARP

provides a transparent and flexible way to integrate users, vehicles, and support-service providers into a single architecture. The agent use ensures the system maintainability, its ability to cope with newer requirements and the possibility to integrate other actors and systems.

The architecture has been tested by implementing three transport planning models and results show that comparable results are obtained. The idea is to continue testing the architecture with diverse scheduling algorithms and negotiation schemes and with a distributed test bed.

References

1. Ambrosino, G. et al, EBusiness Applications to Flexible Transport and Mobility Services. 2001. Available online at: http://citeseer.nj.nec.com/ ambrosino01ebusiness.html.
2. Bellifemine, F. et al, JADE - A FIPA Compliant Agent Framework. C SELT Internal Technical Report, 1999.
3. Bürckert, H; Fischer, K.; et al.. "TeleTruck: A Holonic Fleet Management System", *14th European Meeting on Cybernetics and Systems Research*, 1998, pp 695-700.
4. Coslovich, L.; Pesenti, R.; Ukovich, W. "A Two-Phase Insertion Technique of Unexpected Customers for a Dynamic Dial-a-Ride Problem". Technical Report Working paper, Universita di Trieste, Italy, 2003.
5. Cubillos, C. et al. Multi-Agent Infrastructure for Distributed Planning of Demand-Responsive Passenger Transportation Service. IEEE Int. Conf. on SMC, 2004, pp. 2013 – 2017.
6. Fischer, K.; Müller, J.P.; Pischel, M. Cooperative Transportation Scheduling: An application Domain for DAI. Journal of Applied Artificial Intelligence, Vol. 10, 1996.
7. Horn, M.E.T. "Fleet Scheduling and Dispatching for Demand-Responsive Passenger Services". Transportation Research C, Vol. 10C, 2002, pp. 35-63.
8. Jih, W; Hsu, J. Dynamic Vehicle Routing using Hybrid Genetic Algorithms. IEEE Int. Conf. on robotics & Automation. Detroit, May, 1999.
9. Kikuchi, S.; Donnelly, R.A. "Scheduling Demand-Responsive Transportation Vehicles using Fuzzy-Set Theory". Journal of Transportation Engineering, Vol. 118, No. 3, 1992, pp. 391-409.
10. Kohout, R; Erol, K. Robert C. In-Time Agent-Based Vehicle Routing with a Stochastic Improvement Heuristic. AAAI/IAAI Int. Conf. Orlando, Florida, 1999, pp. 864-869.
11. Miyamoto T.; Nakatyou, K.; Kumagai, S. Route planning method for a dial -a-ride problem, IEEE Int. Conf. on SMC, Vol. 4, 2003, pp. 4002 – 4007.
12. Montemanni, R.; Gambardella, et al. A new algorithm for a dynamic vehicle routing problem based on ant colony system, 2nd Int. Workshop on Freight Transportation and Logistics, 2003.
13. Perugini, D.; Lambert, D.; et al. "A distributed agent approach to global transportation scheduling". IEEE/ WIC Int. Conf. on Intelligent Agent Technology, 2003, pp 18-24.
14. Teodorovic, D.; Radivojevic, G. "A Fuzzy Logic Approach to Dynamic Dial-A-Ride Problem". Fuzzy Sets and Systems, Vol. 116, 2000, pp. 23-33.
15. Uchimura, K. et al. "Demand responsive services in hierarchical public transportation system". IEEE Trans. on Vehicular Technology, Vol. 51, Issue 4, 2002, pp. 760 – 766.
16. Weiss, G., Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, MIT Press, Massachusetts, USA. 1999.